Mechanical simulation of the Exomars rover using Siconos in 3DROV

Vincent Acary, Maurice Brémond, Konstantinos Kapellos, Jan Michalczyk, Roger Pissard-Gibollet

MECHANICAL SIMULATION OF THE EXOMARS ROVER USING SICONOS IN 3DROV

Vincent Acary[1], **Maurice Brémond**[1], **Konstantinos Kapellos**[2], **Jan Michalczyk**[1], **and Roger Pissard-Gibollet**[1]

[1]*INRIA Grenoble, Inovallée, 38330 Montbonnot, France, email:firstname.name@inria.fr*
[2]*TRASYS Space, Terhulpsesteenweg 6c, 1560 Hoeilaart, Belgium. email:firstname.name@trasys.be*

ABSTRACT

This paper presents ongoing work at INRIA in collaboration with TRASYS on the mechanical modelling and simulation of planetary rovers. We focus our presentation on an specific approach used the INRIA SICONOS software. This approach [2] based on the nonsmooth Mechanics theory is dedicated to the modeling and the simulation of multi–body systems with joints, hard contacts, Coulomb's friction and impacts. In addition, software integration aspects are considered defining and implementing generic software interfaces between SICONOS and external 3D tools. The approach is applied to the rover planetary operations enhancing the fidelity and the genericity of the ExoMars existing rover mechanical models used in the TRASYS 3DROV [24] simulator.

Key words: Siconos; 3DRov; contact; impact; Coulomb's friction; nonsmooth dynamics ; numerical simulation;.

1. INTRODUCTION

This paper presents ongoing work at INRIA in collaboration with TRASYS on the mechanical modelling and simulation of planetary rovers. The first aim is to use the modeling and the simulation features of the INRIA SICONOS software dedicated to nonsmooth dynamical systems [2] into a specific rover simulator developed by TRASYS, 3DROV. In this particular application, we focus our work on the simulation of multi-body systems that undergo unilateral contact, friction and impact with its environment. We present a open–source software code, able to deal with idealized contact interaction based on unilateral (rigid) contact, 3D Coulomb friction and impacts. Using such a software into an overall simulation tool as 3DROV may help a lot in designing the control law for locomotion and simulating the global behavior of the rover. In addition, software integration aspects are considered defining and implementing generic software interfaces between SICONOS and external 3D tools. The approach is applied to the rover planetary operations enhancing the fidelity and the genericity of the ExoMars existing rover mechanical models used in the TRASYS 3DROV [24] simulator. The ultimate objective of this work is to integrate SICONOS and 3DROV from the rover mechanical modelling phase to the rover motion simulation and 3D Visualization.

Outline. In Section 2, we briefly review the literature on the modeling and simulation of rovers. In Section 3, we present the main features of Siconos with a brief explanation of the interest of using a nonsmooth approach for contact and friction simulations. Section 3 ends with several stand-alone experiments of obstacle avoidance and traction performance. In Section 4, the TRASYS 3DROV [24] rover planetary operations simulator is presented. Section 5 concludes the paper.

2. STATE OF THE ART

The design of the rover chassis and the study of the locomotion have already been the object of a lot of efforts from several teams in the NASA's Mars Exploration Rovers(MER) project and in the ESA's ExoMars project. The main subjects of interests are the static stability of the rover over steep slopes and the kinematic studies for obstacle avoidance which asks for specific requirements on the geometry of the suspension. The traction performance, which demands for minimal friction requirements has also been extensively studied. To achieve these various goals, designers have worked mainly on the suspension systems and the design of the wheels in order to obtain a good trade-off of performances over various terrains from soft soils (sands, fine granular materials) to rigid surfaces (rocks, coarse granular materials). For an overview of these efforts, we refer to [17, 19, 20, 28] and [22] for a survey.

Physical–based modeling and simulation of planetary rovers is an integral part of planning, testing and design of robotic planetary missions. Therefore, most of these studies have been based on simulation software codes, which are for most of them overall simulation tools containing the simulation of multi-body systems in their environment (mainly with the interaction wheel/terrain), the simulation of the control systems, the hardware in the loop system and the 3D Visualization rendering and animations systems. One of the first piece of software is ROAMS (Rover Analysis, Modeling and Simulation) developed at the Jet Propulsion Laboratory (JPL) for the NASA's MER project. It is based on the DSHELL/DARTS software[32, 15, 14, 8] and offers functionalities for the overall mission simulation with embedded software ans operator–in–the–loop simulator. A particular focus is made on the wheel/terrain interaction by integrating on the Bekker model [6, 7] and its extensions. On the ESA's side, a lot of efforts has been done to develop robust and efficient simulation tools.

Among others, this work includes RCAST developed by MDA and the Dalhousie University in Canada. It is a simulator including 3D simulation of multibody systems from CAD model (SolidWorks and Pro/E) based on Matlab/Simulink toolbox, SimMechanics. It includes rich wheel-soil interaction models developed in a Simulink S-Function blockset based on the AESCO Soft Soil Tire Model (AS^{2}TM enhancement of the Bekker model)[5] and a standard control system based on sensors and actuators in Simulink. This software is embedded in the Simulink's Virtual Reality Toolbox. Other simulators dedicated for the simulation of general multibody systems have also been used like SIMPACK at DLR [11, 19, 20] with an elasto–plastic models with damping of soils and experimental fitting of parameters or COSMOS/Motion in [21] in the context of Rover Mobility Performance Evaluation Tool (RMPET). Recently, an effort has also been done in the framework of 20sim to develop a software for the port–based modeling and simulation of the rover on rough terrain [23, 30]. One of the main advantage of the port–based and the bond graph approach is the possibility to integrate in the same model multi–physical systems and the control systems.

Concerning the specific aspect of the simulation of multi–body systems with contact and friction, all the previous software codes share a wheel/terrain modeling based on some extensions of the Bekker model. From the computational point of view, this model is very convenient because it is sufficiently regular to simulate it with any kind of numerical time integration solvers. If Bekker's model is well–suited for soft soil/wheel interactions, it may be more debatable for rigid soils like rocks or coarse granular materials. In this latter case, the stiffness constants at contact that models the rigid contact of the wheel and the rocks drastically increase and put the standard simulation tools in troubles. This disadvantage in terms of modeling and simulation efficiency can be overcome with a nonsmooth approach as we propose in Siconos. Dealing with inequality constraints to model extremely stiff contacts rather than using large stiffness constants ensures the stability and the efficiency of the solvers. Furthermore, the choice of complex models at contact with a lot of parameters can be difficult to use on varying and uneven terrain. The problem of identifying parameters and their large variability can be a drawback for designing and testing robust control laws and performing realistic predictive simulations. Most of the time, when stiff contact are involved, the respect of impenetrability between bodies, the ideal modeling of threshold and multi–valued effects of the Coulomb friction and the coverage of the dissipation effects with an impact law are sufficient and even more robust for the simulation and the analysis of mechanical systems.

In the more general framework of the simulation of robotic systems, commercial pieces of software have also been used to simulate systems in the framework of the ExoMars project. One can cite MapleSim[1] and Mathworks SimMechanics[2]. Other tools as Modelica (vehicle dynamics or 3D mechanical systems toolboxes)[3] or Microsoft Robotics developer Studio[4] may also be used, although as far as we know no publication is available. In 3DRov [24] or more general robotic simulator like Gazebo[5], physical engines that come from computer games and animation like ODE, Bullet[6] or PhysX[7] are also used for mainly two reasons. The first one is that ODE and Bullet are open source software codes and the second one is that they deal with an apparent simplicity with idealized contact and friction models. Nevertheless, since these software codes are developed for the simulation of large scenes of objects for videos games, the realistic Physics and Mechanics are sometimes sacrificed for some performance reasons. For instance, part of the dynamical system may be artificially frozen and the modeling of Coulomb's friction is often not realistic.

3. SICONOS

SICONOS is an Open Source scientific software primarily targeted at modeling and simulating nonsmooth dynamical systems in the most generic sense. In particular, it can deal with the following systems: mechanical systems (Rigid body or solid) with unilateral contact and Coulomb friction as we find in nonsmooth mechanics, contact dynamics or Granular material, switched Electrical Circuit such as electrical circuits with ideal and piecewise linear components Power converter, Rectifier, Phase-locked loop (PLL) or Analog-to-digital converter, sliding mode control systems. Other applications are found in Systems and Control (hybrid systems, differential inclusions, optimal control with state constraints), Optimization (Complementarity systems and Variational inequalities), Biology (Gene regulatory network), Fluid Mechanics and Computer graphics. The software is based on 4 main components :
1. SICONOS/NUMERICS (C API). Collection of low-level algorithms for solving basic Algebra and optimization problems arising in the simulation of nonsmooth dynamical systems: Linear complementarity problems (LCP) Mixed linear complementarity problems (MLCP) Nonlinear complementarity problems (NCP) Quadratic programming problems (QP) Friction-contact problems (2D or 3D) (Second-order cone programming (SOCP)) Primal or Dual Relay problems
2. SICONOS/KERNEL. C++ API that allows one to model and simulate nonsmooth dynamical systems. it contains : Dynamical systems classes (first order one, Lagrangian systems, Newton-Euler systems) and nonsmooth laws (complementarity, Relay, FrictionContact, impact)
3. SICONOS/MECHANICS This component is a modeling toolbox for multi-body systems based on external collision and geometry libraries.
4. SICONOS/FRONT-END generated python bindings for Numerics, Kernel and Mechanics, with a special support for Siconos data structures.

Nonsmooth Mechanics at a glance In this section, we recall the main ingredients of the nonsmooth mechanics

[1] http://www.maplesoft.com/products/maplesim
[2] http://www.mathworks.fr/products/simmechanics/
[3] https://www.modelica.org/libraries_old/ModelicaMultiBody
[4] http://www.microsoft.com/robotics
[5] http://gazebosim.org
[6] http://bulletphysics.org
[7] https://developer.nvidia.com/technologies/physx

approach for the modeling and the simulation of mechanical systems with contact and friction. For more details, we refer to [2]. Let us start with the following Lagrangian formulation of the equations of motion

$$\begin{cases} \dot{q} = v, & \text{(1a)} \\ M(q)\dot{v} + N(q,v) + F(t,q,v,u) = G^{\top}(q)\lambda, & \text{(1b)} \\ \dot{u} = d(t,q,v,u,\lambda), & \text{(1c)} \\ g_k(q) = 0, \quad k \in \mathcal{E} & \text{(1d)} \\ g_k(q) \geq 0, \quad \lambda_k \geq 0, \quad \lambda_k g_k(q) = 0 \quad k \in \mathcal{I} & \text{(1e)} \end{cases}$$

where $q(t) \in I\!R^n$ is a set of coordinates that gives in a unique way the configuration of the system. The vector $v(t) \in I\!R^n$ describes the velocity. The matrix $M \in I\!R^{n\times n}$ is a symmetric, positive and (semi-)definite matrix representing the inertia. The vector of gyroscopic effects is denoted as $N \in I\!R^n$. The vector of forces $F \in I\!R^n$ applied to the system includes the internal and external forces and the effect of the vector $u \in I\!R^{n_u}$ which might be considered as a control input vector. The dynamics of this control can be described by another first order dynamical system (1c). The sets $\mathcal{E} \subset I\!N$ and $\mathcal{I} \subset I\!N$ are the index sets of bilateral constraints ($\mathcal{E}$ stand for equalities) and of *unilateral constraints* ($\mathcal{I}$ stand for inequalities), respectively. The matrix $G(q) = \nabla_q^{\top} g(q)$ is the Jacobian of the constraints. The condition (1e) is called a *complementarity condition*, or equivalently the *Signorini condition*. It can be equivalently written as

$$0 \leq g_k(q) \perp \lambda_k \geq 0 \quad \text{for all } k \in \mathcal{I}. \tag{2}$$

This condition models the basic unilateral contact condition. The complementarity comes from the fact that the reaction force at contact represented by λ can only act when the contact is closed, that is when $g_k(q) = 0$.

In order to complete the model at contact, let us introduce the local contact variables. Usual kinematic relations between the local velocity $U_k \in I\!R^3$ at one contact k and the generalized variables, and by duality, the generalized forces r due to contact forces $R_k \in I\!R^3$ give

$$U_k(t) = G_k(q)\dot{q} = G_k(q)v, \qquad r = G_k^{\top}(q)R_k. \tag{3}$$

Let us drop the subscript k to lighten the notation. By introducing a local frame at contact $(\mathrm{N}, \mathrm{T}, \mathrm{S})$ where N is a unit normal vector, we can decompose the local variables as $U = U_{\mathrm{N}}\mathrm{N} + U_{\mathrm{T}}, U_{\mathrm{T}} \in I\!R^2, \quad R = R_{\mathrm{N}}\mathrm{N} + R_{\mathrm{T}}, R_{\mathrm{T}} \in I\!R^2$. The Lagrangian dynamics in (1) is usually not so smooth. If the normal relative velocity at contact $U_{\mathrm{N}}(t^\star)$ is positive when the bodies hit at time $t^\star$, in other words, when an impact occurs, a velocity jump must occur to satisfy the constraints after $t^\star$ and an impact law has to be added. Let us simply choose the Newton impact law

$$U_{\mathrm{N}}^{+}(t^\star) = -eU_{\mathrm{N}}^{-}(t^\star), \text{ for all } t \text{ such that } g(q(t^\star)) = 0. \tag{4}$$

By formulating the unilateral contact in terms of local variables at contact we get the impact law at the velocity level

$$\text{if } g_{\mathrm{N}} \triangleq g(q) \leq 0, \quad \text{then } 0 \leq U_{\mathrm{N}}^{+} + eU_{\mathrm{N}}^{-} \perp R_{\mathrm{N}} \geq 0. \tag{5}$$

Coulomb's friction can be expressed in a disjunctive form as

$$\text{if } g_{\mathrm{N}} \leq 0, \begin{cases} \text{if } U_{\mathrm{T}} = 0 & \text{then } R \in \mathbf{C} \\ \text{if } U_{\mathrm{T}} \neq 0 & \text{then } R \in \partial\mathbf{C}, \text{and it exists} \\ & a \geq 0 \text{ such that } R_{\mathrm{T}} = -aU_{\mathrm{T}} \end{cases} \tag{6}$$

where $\mathbf{C}$ is the Coulomb friction cone, $\mathbf{C} = \{R, | \ \|R_{\mathrm{T}}\| \leq \mu R_{\mathrm{N}}\}$ with μ the coefficient of friction. Coulomb's friction can also be formulated compactly as a complementarity condition on second–order cones as [2, 4, 3]

$$-\hat{U} \triangleq -\begin{bmatrix} U_{\mathrm{N}} + \mu\|U_{\mathrm{T}}\| \\ U_{\mathrm{T}} \end{bmatrix} \text{ and } \mathbf{C}^* \in \hat{U} \perp R \in \mathbf{C} \tag{7}$$

where $\mathbf{C}^*$ is the dual cone of $\mathbf{C}$ [27].

Interest of the nonsmooth mechanics approach. Here are listed the main interests of using an approach based on nonsmooth Mechanics when we want to deal with multi–body systems with unilateral contacts, friction and impacts:

1. The use of complementarity formulation allows us to deal with hard contacts in a very efficient way avoiding the issues related to the time–integration of stiff problems. When we want to simulate contacts on hard surfaces, avoiding deep inter–penetrations obliges to use large stiffness parameters in a standard approach. This is overcame in the presented approach.
2. Nonsmooth contact laws are based on few parameters (coefficient of restitution and friction). Contact parameters are often very difficult to measure and have a great variability. Proposing contact laws with few parameters that represents the main effects at contact (inter–penetrability avoidance, friction with real sticking phase and plastic dissipation) provides the user with robust simulation tools in view of Control applications. Furthermore, nonsmooth contact laws can be easily extended to take into account more enhanced effects such as stiffness, damping or cohesion[2]. For soft–soil simulation, piecewise continuous Bekker model could be implemented in this context.
3. The associated simulation methods are proven to be efficient on a large number of applications from medium to large scale multi–body systems [26]. Granular materials with large number of contacts points (between 10^4 and 10^6) can be simulated with such algorithms.
4.

Such an approach has been proved to be robust and efficient on several important industrial applications. Siconos is for instance used for more than 5 years in a strong collaboration between Schneider Electric and INRIA for the virtual prototyping of the mechanisms of circuit breakers [1]. The robustness with respect to contact parameters and the efficiency of the simulation that avoids problems due to stiffness provides Schneider Electric with a robust predictive tool for new designs. This numerical methods are also the object of a closed collaboration with ANSYS for the rigid body dynamics toolbox[12].

Architecture and implementation The development team focused on the development of the simulation and

modeling tools which are as most as possible reusable for different scientific contexts.

Any specific pre/post-processing tools has been developed. Dynamical systems and interactions (unilateral constraints) classes provided by Siconos/Kernel may be configured with the help of plugins or by class derivation. For high–level applications SICONOS relies on external software and provides with an efficient computational engine. Concerning the mechanical modeling, it is possible to simulate linear or non-linear rigid or deformable bodies.

The numerical simulation strategies available in Siconos/Kernel are quasi-statics, smooth dynamics, nonsmooth dynamics, each one being either linear or non-linear and simulated with event–capturing or event–detecting schemes. Available time integration schemes are Moreau–Jean scheme, Schatzman–Paoli scheme, nonsmooth Newmark scheme [9, 10], Odepack suite and DAE solvers developed by E. Hairer [13].

Inside SICONOS/KERNEL, the description of the user model relies internally on a primary graph with dynamical systems as nodes and interactions as edges. During time evolution, the determination of active constraints corresponds to the building of sub-graphs of the primary graph. In the case of problems formulated in local coordinates an adjoin graph transformation is dynamically applied to sub graphs in order to drive the computation of the components of the optimization problem given to Numerics solver.

The primary graph structure may be entirely specified once by the user, or it may also be rebuilt during the simulation, as, for example, the result of a contact detection broad-phase. Links with pre/post processing software for collision detection (Bullet contact detection [18]) and CAD libraries (OpenCascasde [29] and PythonOCC [25]) has been set–up. The latter link enables the contact detection between bodies defined by B-Rep (splines, nurbs, ...) and then to compute gaps and local frame at contact. Such a geometrical representation of the data allows to use realistic modeling of surfaces with large sliding. The possible impact laws for mechanics are complementary condition, impact with or without friction.

SICONOS is written in C++ and has an object oriented architecture, there is no graphical user interface (GUI). A simulation is run through a C++ or Python script in which are gathered the objects creations and the method calls on these object that make up the computation. Furthermore, not only the coarse level of command driving is available, but the time loop can be exploded in every step to allow for an interactive simulation. In the same way the whole database is accessible through copy or reference, allowing for a efficient access and aimed at designing any post-processing functions.

More details can be found at `http://siconos.gforge.inria.fr`.

3.1. Rover simulation using Siconos Standalone

The Rover simulation in the Siconos platform allows us to simulate the robot motion taking into account rigorously dynamics and non-regular models of contact friction. Roughly, coding a Siconos application consists in two main steps : the first is to describe the dynamic Rover model and the contact model interaction between wheels and the environment; the second is to define appropriate numerical parameters for the model.

More precisely, any Siconos code (C++ or Python) is composed of the following sub-steps:

1. define some `DynamicalSystem` instances
2. define some `Interaction` instances, and for each `Interaction`:
   define `Relation`
   define `NonSmoothLaw`
3. build a `NonSmoothDynamicalSystem` : all `DynamicalSystem` + all `Interaction` objects arranged into a directed `Graph`
4. build a `Model` that holds the `NonSmoothDynamicalSystem`.
5. define a `Simulation`: the way the behavior of the `NonSmoothDynamicalSystem` will be computed
   define a `TimeDiscretisation`
   choose a strategy: `TimeStepping` or `EventDriven`
   define some integrators for the DynamicalSystems (`OneStepIntegrator`)
6. define a way to formalize and solve the possible non smooth problems (formulation + solver, the `OneStepNSProblem`)
7. associate the `Simulation` to the Model
8. launch the `Simulation` (time loop)

Modeling The Rover dynamical model is a LagrangianDS siconos object (lagrangian formulation). For 3D Rover modeling, the inertia matrix and contact model interaction relations are difficult to set-up by hands. We use a Maple™code to automatically generate these relations, and to export into C files. In Siconos, we import these relations as a plug-in function that are dynamically loaded. The HuMAns toolbox([31]) using Maple™software tool enable us to easily do complex modeling easily with high accuracy.

Originally developed for humanoid robotics research, the HuMAnS toolbox (for Humanoid Motion Analysis and Simulation) proposes a wealth of state - of - the - art algorithms from the field of robotics research for the modeling, the analysis and the simulation. In our case, we used it to generate the multi-body dynamic model for the Rover. For this, the user need to describe the Rover parameters through several Maple™input files. One file must describe kinematic parameters of the Rover, it indicates the relative location of key-points of the model and the degree of freedom of every connecting point. An other file describes the dynamical parameters, it describes the inertials parameters: the gravity vector, the mass of the segment, the position of the segments centers of mass and the inertia matrix of these segments relative to the centers of their attached frames.

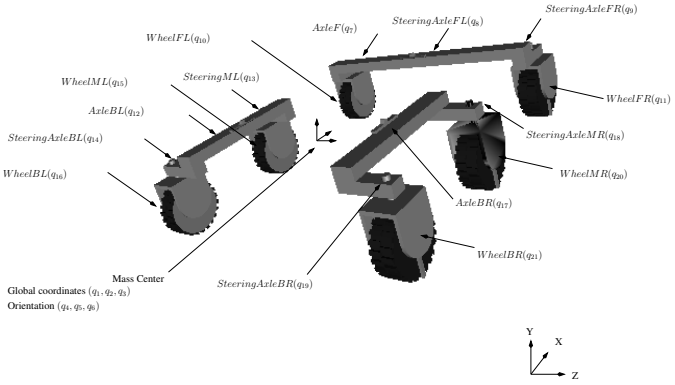


Figure 1: Rover Siconos 3D representation and joints notation

Then, the Maple™generator included in HuMAns produces C code for $M(q)$ the system inertia, $G(q)$ the forces which derived from a potential energy (for example the gravity), $N(q, \dot{q})$ a non-linear effects matrix.

For the simulation we used a simplified model inspired from the ExoMars rover chassis with six wheels on three simple passive bogies located forward at each side of the rover and one on the rear (see [19]). The payload is simply modelised with a mass uniform parallelepiped. The visual representation and joints notation are given on the figure 1. The rover has 21 Degree Of Freeedom (DOF) : 6 for the location of the rover, 6 for the rotation of each wheel, 6 for each wheel steering and 3 for each passive joint of the bogies.

To describe the contact of the Rover with the solid ground or granular soil in Siconos, we need two plugin functions: h to compute the distance between contact points and G which is the transformation matrix of velocity of contact points from the global frame to the local frame.

To simplify the model, we only consider point contact. In this contact model, we simplify the axis of the wheel into a point, and create contact model between this point and the sphere, a plane or a mesh.

For the simulation results given below, the ground is modeled by meshes. In order to be able to perform a simulation of the rover roaming on a triangular mesh in Siconos one needs to have the appropriate interaction and relation classes defined. Relation class is an interface providing distances between the surfaces being checked for collision as well as jacobians, which in turn provide mappings from the local to global coordinate frames. While distances are needed for collision detection, jacobians are crucial for resolving impacts and non-smooth friction problems. Distances calculation is a broad topic in itself, briefly speaking these can be delivered by a third-party collision detection library or calculated according to an explicit algorithm built-in in the respective relation class. In the standalone simulation the latter approach has been chosen. Namely, distance function has been defined explicitly in order to calculate the gap between the center of each wheel of the robot and a triangle arbitrarily oriented in space. Everytime this function returns zero (with certain tolerance) a collision is declared. Jacobian is defined within the relation class as an operator resulting from the multiplication of the local triangle's rotation matrix with an inverse of robot's geometric jacobian. Relation object together with a non-smooth law define the interaction between systems. In the standalone simulation an interaction is instantiated between each wheel and each triangle. Thus, there is six times as many interactions as wheels. This approach can become inefficient as the number of triangles (and complexity of the mesh) grow. To compensate for this problem a space filter should be used in the future which in essence filters out and discards interactions between objects which are far enough.

The C code generated, for the rover model and the contact model interaction, is then imported in Siconos code plugin function. It enables us to easily do complex modeling with high accuracy. This work of modeling is detailed in the report [16].

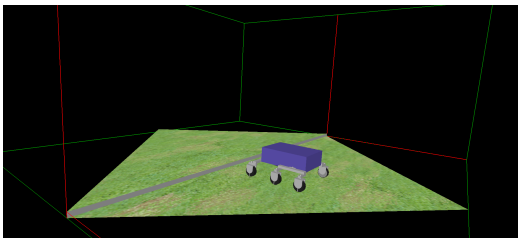
Simulation Results After the modeling step, we describe the simulation strategy and its parameters. For the simulations presented below, we instance a `NewtonImpactNonSmootLlaw` to simulate contact, `Moreau`'s Time Stepping scheme and Newton/AlartCurnier solver to solve the 3D friction problem. Details on the simulation parametrizations can be found on [2]. On the Siconos platform, the 3-D visualization is done in a VRML environment. We simulate the rover with differents initial condition and different environment.

In the sequel, several results are presented obtained from the standalone simulation of the rover in Siconos. Two cases are studied which corresponds to the main tests for the design and analysis the chassis:

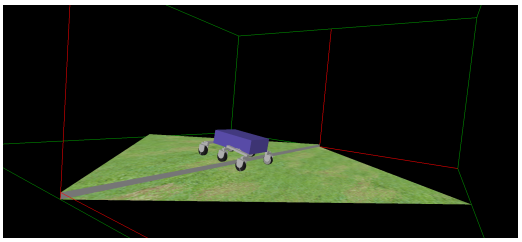
1. Obstacles overcoming: in our experiment, we test the ability to go over an obstacle in the form of a step of 0.1 meter high
2. traction performance: in this latter experiment, we test the ability of the rover to drive uphill with changing wheels' torques and a given coefficient of friction.

Obstacles overcoming In this test case, a 0.1 meter high step is set in front of the robot (Fig. 2). Several impacts occur when robot climbs the step. First impact is due to the fact that rover falls down onto the plane at the beginning of the simulation. Following impacts affect not only y velocity component but also x and z velocity components and are due to the collision with the step. The restitution coefficient equals to 0.1 and the friction coefficient is equal to 0.35. Abrupt changes are observed in velocity of the robot's centre of mass due to impacts.

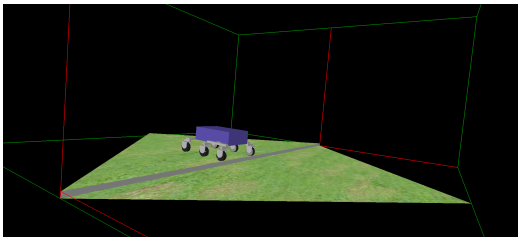
Traction perfomance In this second test case the rover is moving up an inclined plane along the x axis of $2.10^{-}2\,\mathrm{rad}$. The coefficient of friction coefficient is chosen sufficiently low to exhibit sliding under high torques. Six phases of motion have been distinguished in order to depict the various phenomena ans scenarios due to friction:



(a) Phase 1



(b) Phase 2



(c) Phase 3

Figure 2: Rover climbing the step

- no torque is applied to any of the wheels - time interval $0s \leq t < t_1$ with $t_1 = 50s$
- a constant torque counterbalancing the effect of the gravity is applied to all wheels - time interval $t_1 \leq t < t_2$ with $t_2 = 100s$
- a linearly increasing torque is applied to all wheels to stop the rover - time interval $t_2 \leq t < t_3$ with $t_3 = 138.61s$
- a constant torque counterbalancing the effect of the gravity is applied to all wheels - time interval $t_3 \leq t < t_4$ with $t_4 = 180s$
- a linearly increasing torque is applied to all wheels to stop the rover - time interval $t_4 \leq t < t_5$ with $t_5 = 200s$
- a high constant torque is applied to all wheels - time interval $t_5 \leq t$ in order to produce sliding.

In Figure 3, one can see the evolution of rover's mass centre velocity. In the first phase ($[t_0, t_1]$), the rover is rolling without sliding under gravity load. In the second phase the velocity is kept constant due to the driving torque that counteracts the gravity. In the third phase ($[t_2, t_3]$), the velocity is decreasing in quadratic way with respect to time since the torque is linearly increasing. In the fourth ($[t_3, t_4]$), the rover is stopped at rest in the slope. The fifth phase is similar to the third one. One can notice that in the last phase of motion despite dramatic change in wheels' velocities rover's trunk does not accelerate anymore. This

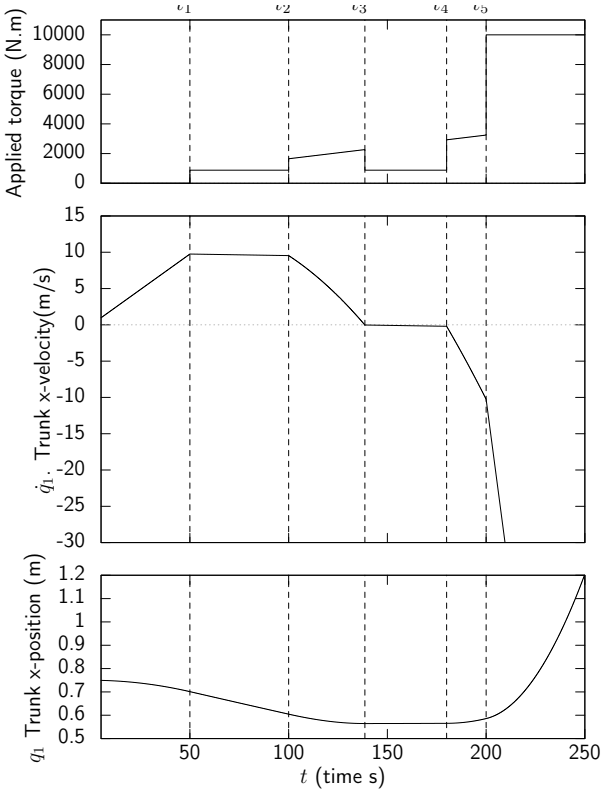


Figure 3: Motion of the rover in a slope under the action of varying driving torques.

is due to the friction force saturation. Energy supplied to the wheels is dissipated by friction and sliding.

One of the most important feature to note is that the ideal Coulomb's law is perfectly satisfied. In the case of a regularized model, some high frequency dynamics would have been observed and the perfect zero relative velocity at contact is never obtained.

4. 3DROV

3DROV is a rover planetary exploration simulation framework developed by TRASYS. It uses as simulation framework the ESOC EGOS SimSat v4 tool[8] and includes both, models of the environment in which the rover evolves (terrain, atmosphere, orbital and timekeeping) and models of the rover itself (mechanical, power, thermal, communications, payloads and a model of the o/b controller). 3DROV disposes also a 3D Visualization component that allows monitoring in a photo-realistic synthetic environment the evolution of the simulation. Thanks to the use of the SMP (Simulation Model Portability) the system provides the possibility to easily use models of different levels of fidelity depending on the scope of the simulation. 3DROV focuses on operational simulations for the validation of rover plans (composition of rover Activities) spanning over one or several sols (experiment cycles). When planning rover operations including rover travelling, it is very important to be able

[8] http://www.egos.esa.int/portal/egos-web/products/Simulators/SIMSAT/

to simulate this part of the plan as close as possible in order to predict the Rover mechanical behaviour, to estimate the path to be travelled and to evaluate the required energy consumption. 3DROV uses the PhysX [9] physics engine and recently Bullet [10] for mechanical simulation. Internal ESA work performed at ESTEC showed that using PhysX the simulation results are qualitatively consistent with the expected behaviour but still insufficient to support detailed rover motion validation. The use of SICONOS allows to involve better friction-contact model on a granular terrain and aims to improve the existing 3DROV simulation framework providing accurate mechanical rover simulation. In addition, this work contributes to the openness of the system by establishing a generic API for dynamics models/3D tools interaction.

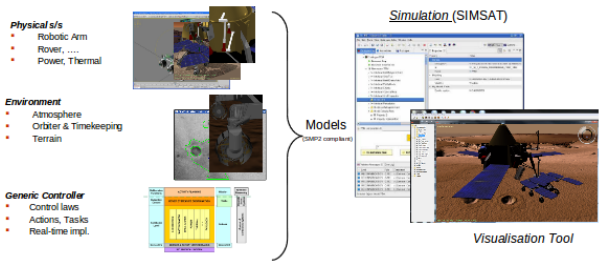


Figure 4: 3DROV architecture

Siconos within 3DROV consists of several shared objects grouped inside the kernel and numerics shared libraries. It is integrated into the Rover Mechanical model component of 3DROV and interacts with a) the on-board controller model to receive the required trajectory and b) with the 3D Visualisation tool to exchange collision information and the rover state. Figure 5 illustrates the Exomars rover model into a test environment and on a planetary terrain. The simulation 'step' is triggered by the simulation framework scheduler.

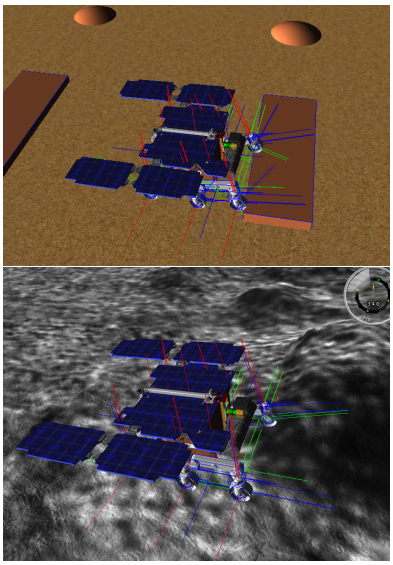


Figure 5: ExoMars simulation in 3DROV

In particular, a dedicated API allows Siconos a) to retrieve from the 3D Visualization component collision information including the number of contact points, coordinates of contact points, normal vector in the contact point, penetration as well as coefficients of friction and restitution and b) to set the rover's joints and the rover's trunk position and orientation.

[9] https://developer.nvidia.com/technologies/physx

[10] http://bulletphysics.org

5. CONCLUSION

3DROV is a rover planetary exploration simulation framework which allows different levels of fidelity depending on the scope of the simulation. Plugging SICONOS allows to involve better friction-contact modeling in view of robust and efficient simulations on ground or with rigid obstacles and aims to improve the existing 3DROV simulation framework providing accurate mechanical rover simulation.

We claim that handling this and that in the context of non-smooth dynamical systems provides, in particular in the case of hard contacts, more representative results and simplifies the modelling of the wheel/terrain interaction.

Next steps of this on-going work includes:

- benchmarking of SICONOS simulation results for static stability of the rover over steep slopes, slippage handling and obstacles overcoming.
- comparison with results obtained using the usual physical engines like Bullet, PhysX and ODE.
- simulation of more complex wheel/terrain interaction taking into account granular materials or the Bekker model.
- continue the implementation of the 3DROV/SICONOS API focussing on openness and genericity both at model description level and information exchange during simulation.

REFERENCES

[1] V. Acary. Projected event-capturing time-stepping schemes for nonsmooth mechanical systems with unilateral contact and coulomb's friction. *Computer Methods in Applied Mechanics and Engineering*, 256:224 – 250, 2013.

[2] V. Acary and B. Brogliato. *Numerical methods for nonsmooth dynamical systems: applications in mechanics and electronics*, volume 35. Springer, 2008.

[3] V. Acary and F. Cadoux. *Recent Advances in Contact Mechanics, Stavroulakis, Georgios E. (Ed.)*, volume 56 of *Lecture Notes in Applied and Computational Mechanics*, chapter Applications of an existence result for the Coulomb friction problem. Springer Verlag, 2013.

[4] V. Acary, F. Cadoux, C. Lemaréchal, and J. Malick. A formulation of the linear discrete Coulomb friction problem via convex optimization. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 91(2):155–175, 2011.

[5] AESCO. Matlab/Simulink module AESCO soft soil tyre model (AS^{2}TM) users guide, 2003.

[6] M. G. Bekker. *Introduction to terrain vehicle systems*. University of Michigan Press, Ann Arbor, USA, 1969.

[7] M. G. Bekker. The development of a moon rover. *Journal of the British Interplanetary Society*, 38(537–543), 1985.

[8] J. Cameron, A. Jain, t. Huntsberger, G. Sohl, and R. Mukherjee. Vehicle-terrain interaction modeling and validation for planetary rovers. Technical Report 10-15, National Aeronautics and Space Administration. Jet Propulsion Laboratory. California Institue of Technology, Pasadena, California, 2009.

[9] Q. Z. Chen, V. Acary, G. Virlez, and O. Brüls. A Newmark-Type Integrator for Flexible Systems Considering Nonsmooth Unilateral Constraints. In Peter Eberhard, editor, *The Second Joint International Conference on Multibody System Dynamics - IMSD 2012*, Stuttgart, Germany, March 2012.

[10] Q.-Z. Chen, V. Acary, G. Virlez, and O. Brüls. A nonsmooth generalized-α scheme for flexible multibody systems with unilateral constraints. *International Journal for Numerical Methods in Engineering*, 2012. submitted.

[11] Andreas Gibbesch and Bernd Schäfer. Multibody system modelling and simulation of planetary rover mobility on soft terrain. In *Proc. of 8th Int. Symp. on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS*, 2005.

[12] M. Haddouni, V. Acary, and J.D. Beley. Comparison of index-3, index-2 and index-1 dae solvers for nonsmooth multibody systems with unilateral or bilateral constraints. In *Eccomas. MultiBody dynamics 2013.*, July 2013.

[13] E. Hairer, S.P. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problems.* Springer, 1993.

[14] A. Jain, J. Balaram, J. Cameron, J. Guineau, C. Lim, M. Pomerantz, and G. Sohl. Recent developments in the roams planetary rover simulation environment. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 2, pages 861–876 Vol.2, 2004.

[15] A. Jain, J. Guineau, C. Lim, W. Lincoln, M. Pomerantz, G. Sohl, and R. Steele. Roams: Planetary surface rover simulation environment. In *in International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2003*, 2003.

[16] supervisor: Vincent Acary Jianhui Yang. Simulation of a mars rover on granular soils. Technical report, INRIA, Erasmus Mundus Master Internship, 2010.

[17] C Lee, J Dalcolmo, S Klinkner, L Richter, G Terrien, A Krebs, R Siegwart, L Waugh, and C Draper. Design and manufacture of a full size breadboard exomars rover chassis. In *Proceedings of 9th ESA workshop on advanced space technologies for robotics and automation, Noordwijk, the Netherlands: Astrionic Press*, pages 28–39. Citeseer, 2006.

[18] Bullet Physics Library. `http://bulletphysics.org`.

[19] S Michaud, A Gibbesch, T Thueer, A Krebs, C Lee, B Despont, B Schäfer, and R Slade. Development of the exomars chassis and locomotion subsystem. In *9th International Symposium on Artificial Intelligence. Universal City, CA,, USA: Robotics and Automation for Space (i-SAIRAS 2008)*, 2008.

[20] S Michaud, M Hoepflinger, T Thueer, C Lee, A Krebs, B Despont, A Gibbesch, and L Richter. Lesson learned from exomars locomotion system test campaign. In *Proceedings of 10th Workshop on Advanced Space Technologies for Robotics and Automation, ESTEC The Netherlands*, 2008.

[21] N. Patel, A. Ellery, E. Allouis, M. Sweeting, and L. Richter. Rover mobility performance evaluation tool (rmpet): A systematic tool for rover chassis evaluation via application of bekker theory. In *8th ESA Workshop on Advanced Space Technologies for Robotics and Automation, ASTRA 2004*, pages 251–258. ESTEC, Noordwijk, The Netherlands, 2-4 November 2004.

[22] Nildeep Patel, Richard Slade, and Jim Clemmet. The exomars rover locomotion subsystem. *Journal of Terramechanics*, 47(4):227–242, 2010.

[23] P. Poulakis, L Joudrier, and S Stramigioli. Port-based modeling and simulation of planetary rover locomotion on rough terrain. In *9th ESA Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA'06), Noordwijk, The Netherlands*. Citeseer, 2006.

[24] P. Poulakis, L. Joudrier, S. Wailliez, and K.Kapellos. 3drov: A planetary rover system design, simulation and verification tool. In *10th International Symposium on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS-08*, Hollywood,USA, February 26-29 2008.

[25] PythonOCC. 3D CAD/CAE/PLM development framework for the Python programming language. `http://www.pythonocc.org`.

[26] F. Radjaï and F. Dubois, editors. *Discrete–element modeling of granular materials*. Iste. John Wiley & Sons, 2011.

[27] R.T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[28] T. Oechslin S. Michaud, P. Oettershagen. Wheel level test data generation and utilization to predict locomotion performances of planetary rovers and validate simulation tools. In *Proceedings of International Symposium on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS*, 2012.

[29] Open CASCADE Technology. `http://www.opencascade.org`.

[30] Martin Wassink, Raffaella Carloni, Pantelis Poulakis, and Stefano Stramigioli. Digital elevation map reconstruction for port-based dynamic simulation of contacts on irregular surfaces. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 5179–5184. IEEE, 2009.

[31] P.-B. Wieber, F. Billet, R. Boissieux, L.and Pissard-Gibollet, et al. The humans toolbox, a homogenous framework for motion capture, analysis and simulation. In *International Symposium on the 3D Analysis of Human Movement*, 2006.

[32] J. Yen, A. Jain, and J. Balaram. Roams: Rover analysis, modeling and simulation. In *In Proceedings of the 1999 International Symposium on Art Intelligence, Robotics and Automation for Space*, pages 249–254, 1999.